The Wisconsin PASS Project

Andrea Arpaci-Dusseau, Remzi Arpaci-Dusseau,
Ben Liblit, Miron Livny, Michael Swift

University of Wisconsin, Madison

Storage System Reliability

Goals

Study: How disks fail

- Latent sector errors [Sigmetrics 07]
- Corruption [FAST 08]

Study: How & why systems fail

- Model checking [FAST 08] ←
- Pointer corruption [DSN 08]
- Error propagation [FAST 08] ←

Develop: New ways to cope with failure

- I/O Shepherding [SOSP 07]
- Declarative FS Checking [OSDI 08] ←

Storage Design
Model Checking

The Problem

RAID protection schemes are complex

- Identity info (logical, physical)
- Block, sector, parental checksums
- Disk scrubbing
- Write verify

Systems use different combos of above

- Systems from NetApp, Dell, Hitachi, etc.

What does a given scheme protect?

Model Checking

Automatic checking

- Use search to see how designs react to different types of failure

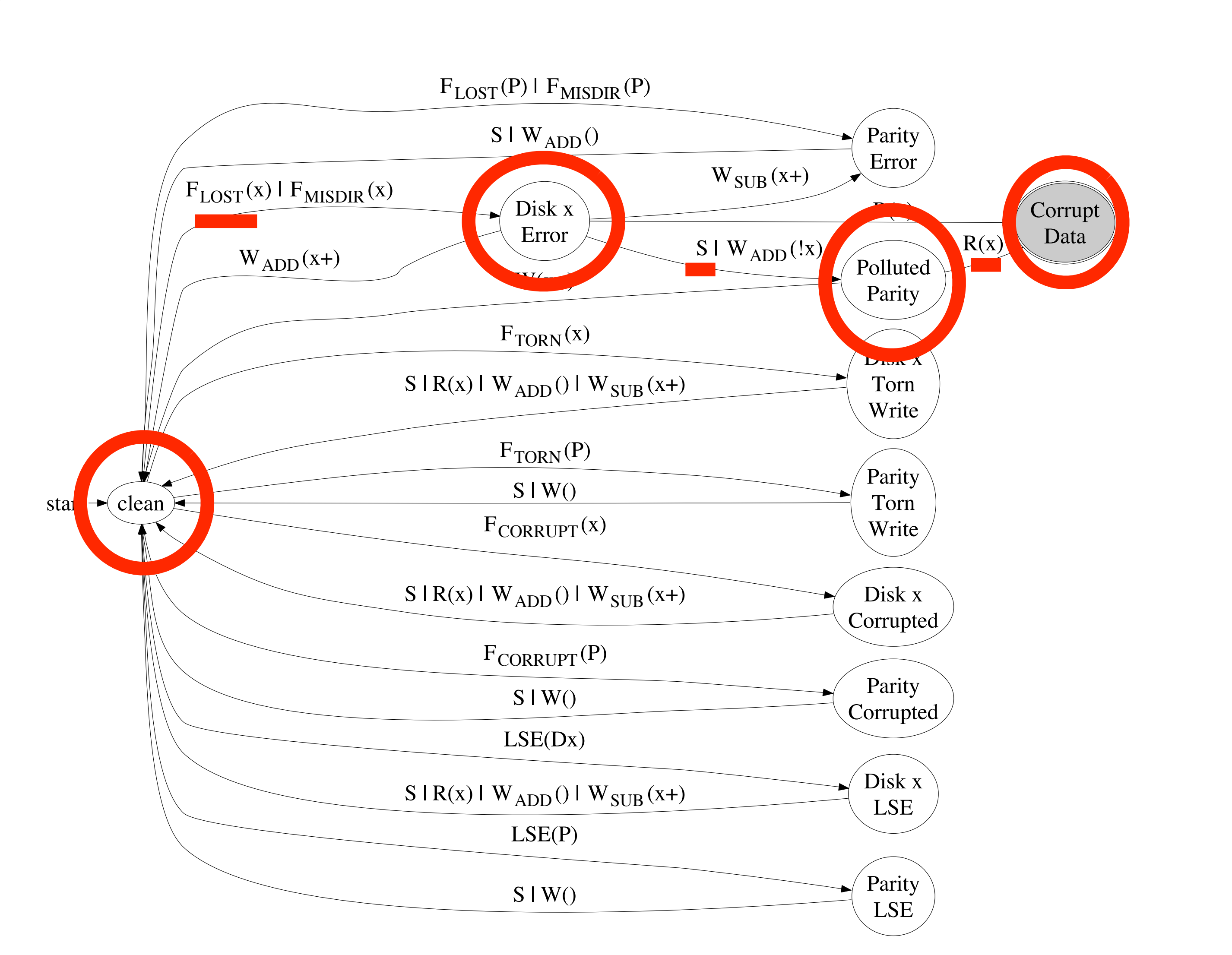
Example faults

- Lost writes
- Torn writes
- Corruption

Output:

- “Proof” of design correctness or example of hole in scheme

Example: Block checksums + Scrubbing



Results [details in FAST ’08]

Analyzed eight real designs

- From industry and literature

Found design flaws in all eight

- Wrong sequence of faults leads to data loss or unavailability
- General problem: Parity Pollution

New technique: Version mirroring

- Overcomes pollution thru additional state

Error Propagation

The Problem

Problem: Lost Errors

- Low-level generates error (EIO)
- Somewhere before it gets reported, file system loses the error

Causes many problems

- Can’t tell if operation worked
- File system itself can’t detect/recover

How to find where these occur?

The Approach: EDP

Static source code analysis

- Look through source code for places where error codes are lost

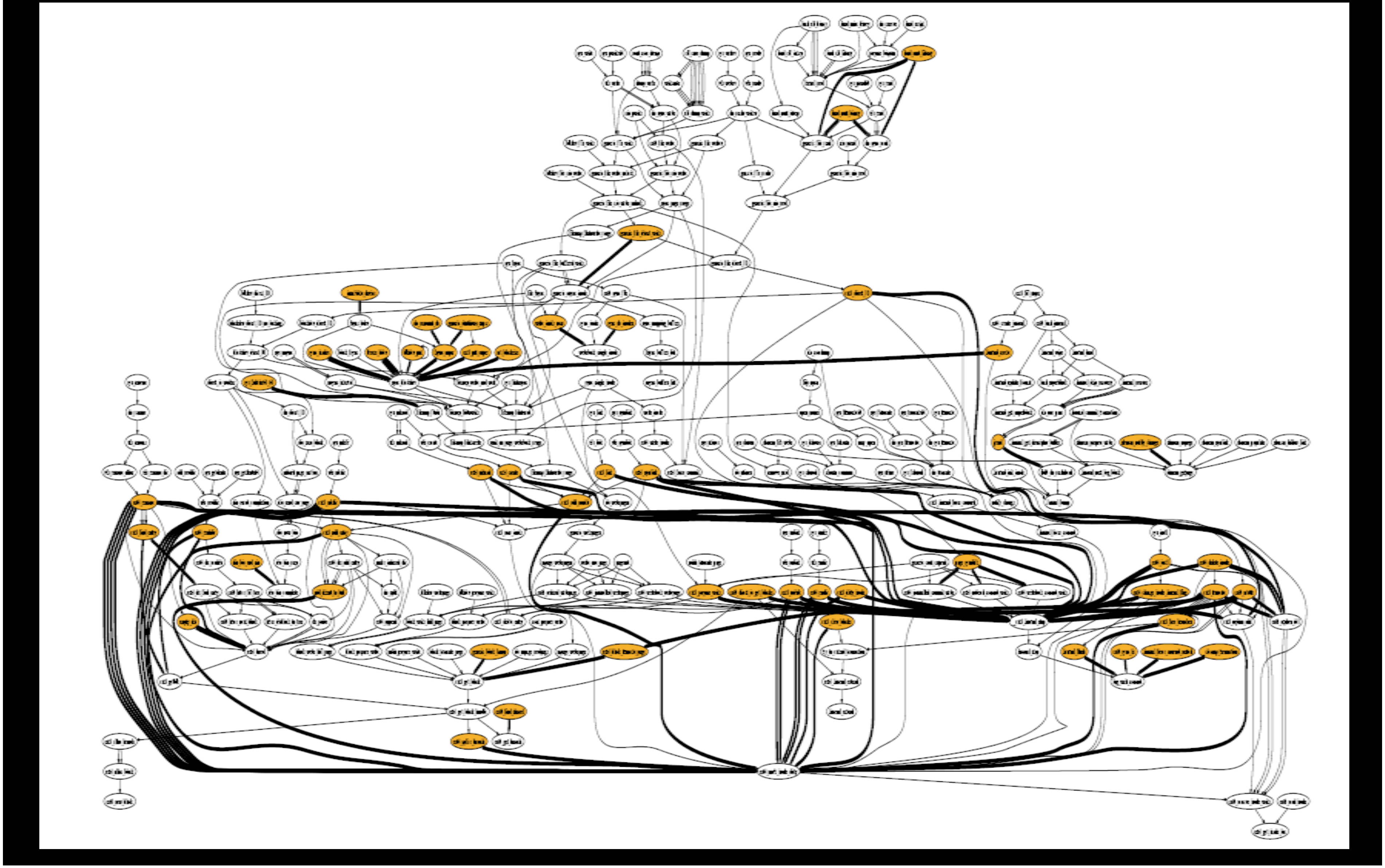
Built in CIL framework (from UCB)

- Error generation: Return value or arg
- Constructs channels:
  Flow of errors back through call graph
- Label channel complete or broken

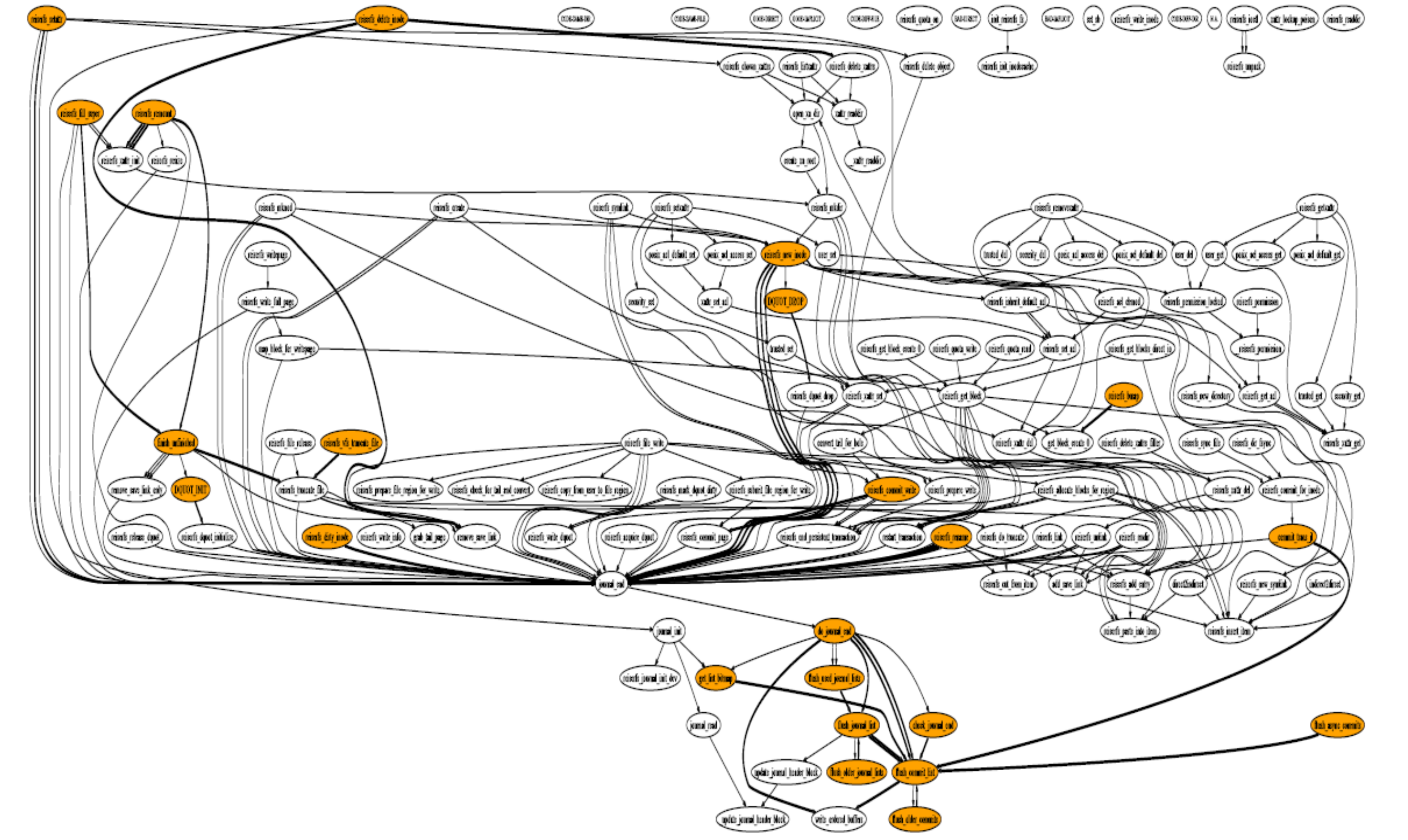
Example

```
int sync_blockdev (block_device *) {
    int ret = 0, err;
    ret = filemap_fdatawrite();
    err = filemap_fdatawait();
    if (!ret) {
       ret = err;
       return ret;
    }
  }
int journal_recover (journal *) {
    int err;
    …
    sync_blockdev(); // ERROR IGNORED!
    …
    return err;
  }
```

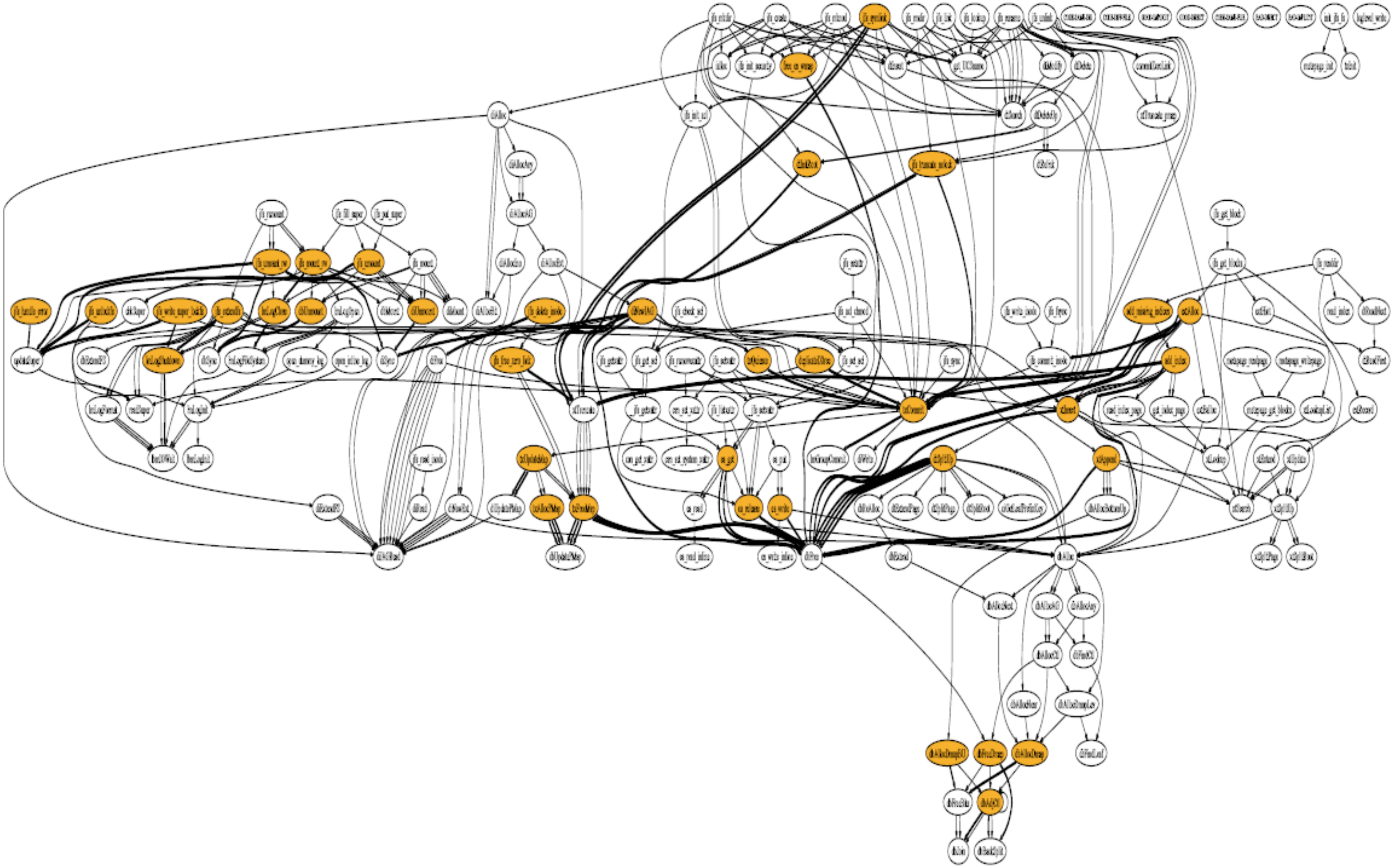
ext3



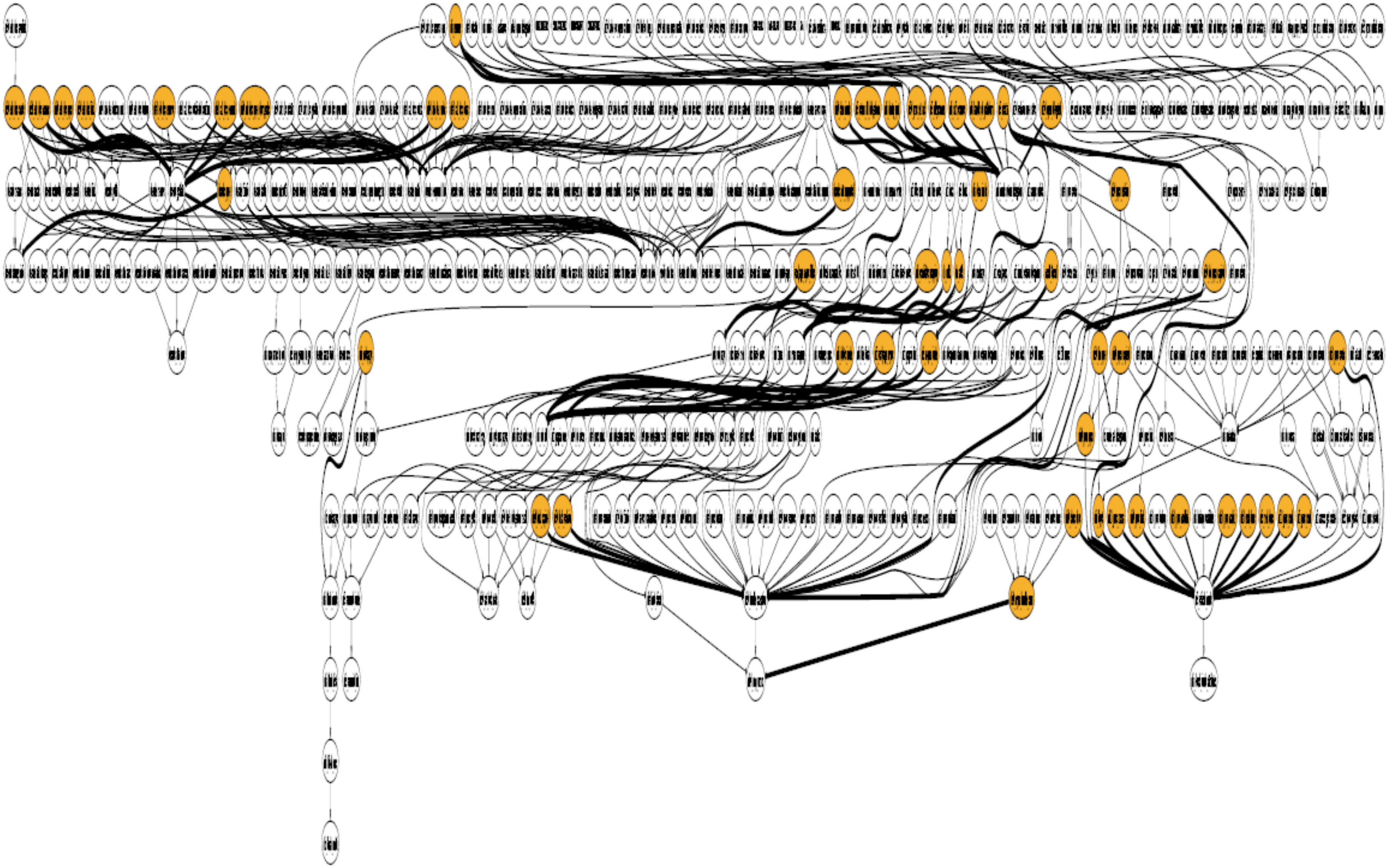
reiserfs



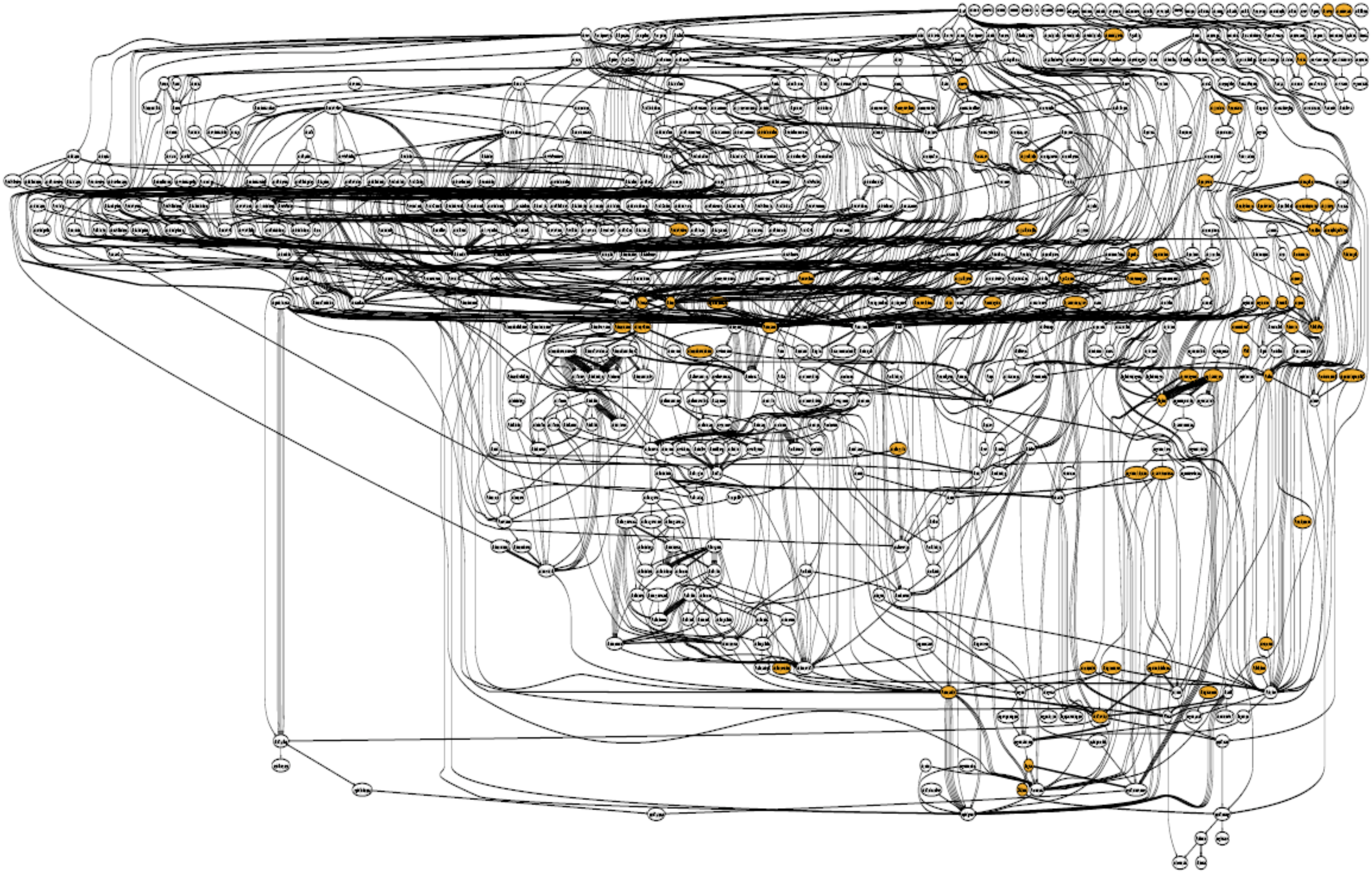
IBM JFS



NFS Client



SGI XFS



Summary [details in FAST ’08]

Static Analysis

- Show how basic error codes flow
- Currently: 34 basic error codes

Target systems

- 51 Linux file systems
- 3 storage drivers

Results

- Roughly 10k function calls
- More than 10% of calls are problematic

Declarative File System Checking

The Problem

Too much C code

- As long as we build systems this way, they will always have bugs

Different approach: Higher-level

- Fewer lines of code to describe system

Target domain: File system checking

- Important and complex code (tens of thousands of lines of C code)

Our Approach: SQCK

SQCK (“squeak” for SQL-based FSCK)

- Load file system metadata into tables
- Run SQL queries over metadata, perhaps fixing problems
- Store metadata back into file system

Lots of challenges

- Simplicity
- Performance
- Flexibility

Example Query

Check for out-of-range indirect block

```
SELECT X.*
FROM   ExtentTabl X, SuperBlkTabl S
WHERE  S.copyNum=1              AND
       X.Type=INDIRECT_POINTER  AND
       (X.Start < S.firstBlock OR
        X.End >= S.lastBlock)
```

Results [details in OSDI ’08]

Simplicity

- Roughly 1k lines of SQL to implement most of e2fsck
- Ordering of checks/repairs an issue

Performance

- Comparable to e2fsck (usually)

Flexibility

- Can change policies more easily (it’s declarative!)

Impact

Impact

Publications

Open-source

- Found hundreds of bugs in ext, Reiser, XFS, and other file systems
- Many now fixed (some during presentation!)
- Some design influence on next-gen FS’s

Industrial

- Led to re-design of storage protection scheme in RAIDs

People

Current students

- Lakshmi Bairavasundaram (Ph.D. --> NetApp)
- Haryadi Gunawi (Ph.D.)
- Cindy Rubio (Ph.D.)
- Abhishek Rajimwale (Ph.D., maybe)
- Andrew Krioukov (Undergraduate --> Berkeley)
- Nitin Agarwal (Ph.D.)

Former students

- Meenali Rungta (Google), Shweta Krishnan (Cisco), Arini Balakrishnan (Sun)

Collaboration with many@NetApp, Schroeder@Toronto

Conclude

Conclusions

Analysis of disk failures

- Good first steps
- New devices on horizon

Analysis of systems

- Everywhere we look there are problems
- Starting to understand how to measure reliability (model checking, static analysis)
- Still hard to do this in a general way

Building reliability in by design

- Just the beginning (SQCK)

Want to read more?

www.cs.wisc.edu/adsl

Thanks to our sponsors:

NSF

NetApp
Go further, faster